Efficient Two-Phase Approaches for Branch-and-Bound Style Resource Constrained Scheduling

Mingsong Chen*, Fan Gu, Lei Zhou, Geguang Pu and Xiao Liu

* Presenter

January 7, 2014
Outline

- Introduction
- RCS using Branch-and-Bound Approaches
  - Graph–based Notations
  - BULB Approach
- Our Two-Phase Approaches
  - Bounded-Operation Approach
  - Non-Chronological Backtrack
  - Search Space Speculation
- Experiments
- Conclusion
Outline

- **Introduction**
- **RCS using Branch-and-Bound Approaches**
  - Graph–based Notations
  - BULB Approach
- **Our Two-Phase Approaches**
  - Bounded-Operation Approach
  - Non-Chronological Backtrack
  - Search Space Speculation
- **Experiments**
- **Conclusion**
SoC Design Cost Model

Big Savings by using ESL Methodology

Rising cost of IC design and effect of CAD tools
(Courtesy: Andrew Kahng, UCSD and SRC)
High Level Synthesis

- Convert ESL specification to RTL implementation, and satisfy the design constraints.
  - **Input:** Behavior specifications (C, SystemC, etc.), and design constraints (delay, power, area, etc.)
  - **Output:** RTL implementations (datapath, controller)

```plaintext
int Sample()
{
    var A, B, C, D, E, F, G : int;
    Read(A, B, C, D, E);
    F = E * (A + B);
    G = (A + B) * (C + D);
    ....
}
```
Resource Constrained Scheduling

- **Scheduling** is a mapping of operations to control steps
  - Given a DFG and a set of resource constraints, RCS tries to find an (optimal) schedule with minimum overall control steps.
- Various resource constraints (e.g., functional units, power, ...).

RCS is NP-Complete. RCS should take care of
1) Operation precedence. 2) Resource sharing constraints

Constraints:
- Delay(+) = 1,
- Delay(*) = 2,
- Functional units: 1+, 1*

Schedule length = 8
Basic Solutions

- **Non-optimal heuristics**
  - Force Directed Scheduling
  - List scheduling
    - Pros: Fast to get near-optimal results
    - Cons: schedules may not be tight

- **Optimal sequential approaches**
  - Integer linear programming (ILP)
    - Pros: easy modeling
    - Cons: scalability, cannot handle non-integer time
  - Branch-and-bound
    - Pros: can prune the fruitless search space efficiently
    - Cons: hard to achieve a tight initial upper-bound
Outline

• Introduction

• RCS using Branch-and-Bound Approaches
  ◆ Graph–based Notations
  ◆ BULB Approach

• Our Two-Phase Approaches
  ◆ Bounded-Operation Approach
  ◆ Non-Chronological Backtrack
  ◆ Search Space Speculation

• Experiments

• Conclusion
Based on [ASAP, ALAP], naively enumerating all the possibilities can be extremely time consuming

- The operations are enumerated in a specific order
- Each operation is enumerated from ASAP to ALAP

A schedule is a binary relation of operations and corresponding dispatching control step

- E.g., \{(v1, 1), (v2, 2), (v3, 3), (v4, 5), (v5, 7)\}

Constraints:
- Delay(+) = 1, 1+
- Delay(*) = 2, 1*
BULB tries to prune fruitless enumerations.

B&B approach keeps two data structure regarding bound information.

- $S_{bsf}$, best complete schedule searched so far
- $S$, current incomplete schedule
Pruning in BULB

- Pruning \([\text{lower} > \omega] \rightarrow \text{Backtrack to last operation}\)
- Termination \([\text{globalLow} == \omega \text{ or fully explored}]\)
- Substitution \([\text{if}(\text{upper} < \omega) \omega = \text{upper}]\)

\(\omega\) plays an important role in B&B approaches. A smaller \(\omega\) can
- tighten the [ASAP, ALAP] intervals, i.e., search space;
- enable the fast pruning of inferior schedules during RCS.
Outline

- Introduction
- RCS using Branch-and-Bound Approaches
  - Graph–based Notations
  - BULB Approach
- Our Two-Phase Approaches
  - Bounded-Operation Approach
  - Non-Chronological Backtrack
  - Search Space Speculation
- Experiments
- Conclusion
Importance of Initial Feasible Schedule

- ALAP(OPi, Sinit) = $\omega_{\text{init}} - \text{CPw}(\text{G(OP}_i))$, where Sinit is an initial feasible schedule, and $\omega_{\text{init}} = \text{length}(S_{\text{init}})$.

- List scheduling cannot always guarantee a small $\omega_{\text{init}}$, since it only considers only one possible schedule combination of unscheduled operations.

- How to quickly find a small $\omega_{\text{init}}$ is a key issue in RCS.

$\omega_{\text{init}} = 8$

$\omega'_{\text{init}} = 7$
Basic Idea of Our 2P Approach

- Two-phase approach has two steps
  - Step 1 does **partial search** on the search space coarsely to achieve a tight schedule.
  - Step 2 fully scans the search space in the same way as BULB approach, but with a tight $\omega_{\text{init}}$ achieved from step 1.

- Partial Search should achieve a small $\omega_{\text{init}}$ with small overhead.
Bounded Operation Approach

- Basic idea: **Less operations** involved in partial search.
- Bounded operation approach only considers the input nodes. The remaining nodes are estimated using list scheduling approach.
- Example:

Only 4 tries can achieve the tightest initial schedule. Bounded operation method can efficiently avoid trap in the deep search.
Search Space Speculation

- Basic idea: **Smaller search range** of each operation.
- By adopting a greedy strategy, our speculation approach assumes that the global optimal result will be always located in the first half of original range.

Example:

Only 4 tries can achieve the tightest initial schedule.
Search space speculation can efficiently avoid trap in the deep search.
Non-Chronological Backtrack

- **Basic idea**: A large backtrack jump to escape the local deep search.

- **Our non-chronological partial search is based on the DFG level structure.**
  - **Level** indicates the precedence between operations.
  - **Level check condition**: All the operations in the $i^{th}$ level are scheduled, and for each operation $op_{i,j}$ in the $i^{th}$ level, $S_{bsf}(op_{i,j}) \leq S(op_{i,j})$

$$
\begin{align*}
\text{level1} & : v1 + [1, 3] \rightarrow v2 + [1, 3] \\
\text{level2} & : v3 * [2, 5] \rightarrow v4 * [2, 5] \\
\text{level3} & : v5 * [4, 7]
\end{align*}
$$
Non-Chronological Backtrack

- When level check condition holds in the $i$th level, the scheduling will backtrack to the first dispatched operation of $i$th level.

- Example

$$
\begin{align*}
S_{bsf} &= \{(OP_1, 1), (OP_2, 2), (OP_3, 3), (OP_4, 5), (OP_5, 7)\} \\
S_1 &= \{(OP_1, 1)\} \\
\text{ListScheduling}(S_1) &= 8 \\
S_2 &= \{(OP_1, 1), (OP_2, 2)\} \text{will backtrack due to the level check condition} \\
S_2' &= \{(OP_1, 2), (OP_2, 1)\} \\
\text{ListScheduling}(S_2') &= 7 = \text{Length}(S_{opt})
\end{align*}
$$

Only 2 tries can achieve the tightest initial schedule. Non-chronological backtrack can efficiently escape the deep search.
Outline

- Introduction
- RCS using Branch-and-Bound Approaches
  - Graph–based Notations
  - BULB Approach
- Our Two-Phase Approaches
  - Bounded-Operation Approach
  - Non-Chronological Backtrack
  - Search Space Speculation
- Experiments
- Conclusion
Benchmarks & Settings

- Using benchmarks from *MediaBench*.
- Implementing BULB & our approach using C++.
- All experiments were conducted on a Linux machine with Intel Xeon 3.3GHz Processor and 8G RAM.
- Setting of functional units:

<table>
<thead>
<tr>
<th>Functional Unit</th>
<th>Operation class</th>
<th>Delay (unit)</th>
<th>Power (unit)</th>
<th>Energy (unit)</th>
<th>Area (unit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD/SUB</td>
<td>+/-</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>MUL/DIV</td>
<td>*/</td>
<td>2</td>
<td>20</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>MEM</td>
<td>LD/STR</td>
<td>1</td>
<td>15</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>Shift</td>
<td>&lt;&lt;/&gt;&gt;</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>Others</td>
<td>...</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>
## Results under Functional Constraints

RCS efforts are significantly improved:
- Our 2P approaches outperform both ILP and BULB approaches
- Parallel execution of 2P methods may achieve the best overall performance

<table>
<thead>
<tr>
<th>Design Name</th>
<th>CP</th>
<th>BULB</th>
<th>Bounded $T_1$</th>
<th>Operation $T_{total}$</th>
<th>Non-Chronological $T_1$</th>
<th>Non-Chronological $T_{total}$</th>
<th>Space Speculation $T_1$</th>
<th>Space Speculation $T_{total}$</th>
<th>Max Impr.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARFilter</td>
<td>TO</td>
<td>0.16</td>
<td>0.08</td>
<td>0.22</td>
<td>&lt;0.01</td>
<td>0.16</td>
<td>0.10</td>
<td>0.25</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>0.40</td>
<td>0.13</td>
<td>0.49</td>
<td>&lt;0.01</td>
<td>0.40</td>
<td>0.27</td>
<td>0.62</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>0.38</td>
<td>0.14</td>
<td>0.49</td>
<td>&lt;0.01</td>
<td>0.38</td>
<td>0.26</td>
<td>0.62</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>1.40</td>
<td>0.01</td>
<td>&lt;0.01</td>
<td>0.02</td>
<td>&lt;0.01</td>
<td>0.01</td>
<td>&lt;0.01</td>
<td>0.01</td>
<td>1.00</td>
</tr>
<tr>
<td>Collapse</td>
<td>TO</td>
<td>TO</td>
<td>TO</td>
<td>162.20</td>
<td>TO</td>
<td>TO</td>
<td>0.18</td>
<td>0.18</td>
<td>&gt;2.00e4</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>TO</td>
<td>TO</td>
<td>TO 162.20</td>
<td>TO</td>
<td>TO</td>
<td>NA</td>
<td>159e4</td>
<td></td>
</tr>
<tr>
<td>Cosine 1</td>
<td>TO</td>
<td>63.51</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
<td>0.06</td>
<td>0.06</td>
<td>20.94</td>
<td>20.94</td>
<td>1.59e4</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>377.58</td>
<td>15.66</td>
<td>15.66</td>
<td>0.03</td>
<td>330.23</td>
<td>21.70</td>
<td>21.70</td>
<td>24.11</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
<td>1.00</td>
</tr>
<tr>
<td>FDCT</td>
<td>TO</td>
<td>20.30</td>
<td>0.07</td>
<td>0.07</td>
<td>18.56</td>
<td>18.56</td>
<td>0.26</td>
<td>0.26</td>
<td>5.08e3</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>113.92</td>
<td>0.07</td>
<td>0.07</td>
<td>19.87</td>
<td>19.87</td>
<td>0.65</td>
<td>0.65</td>
<td>1.63e3</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>11.17</td>
<td>0.63</td>
<td>0.63</td>
<td>1.94</td>
<td>1.94</td>
<td>2.03</td>
<td>2.03</td>
<td>17.73</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>2.49</td>
<td>0.03</td>
<td>0.03</td>
<td>0.23</td>
<td>0.23</td>
<td>3.76</td>
<td>3.76</td>
<td>83.00</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>0.48</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
<td>0.12</td>
<td>0.12</td>
<td>0.16</td>
<td>0.16</td>
<td>120.00</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>0.34</td>
<td>0.21</td>
<td>0.21</td>
<td>0.03</td>
<td>0.03</td>
<td>0.14</td>
<td>0.14</td>
<td>11.33</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>0.07</td>
<td>0.01</td>
<td>0.01</td>
<td>0.06</td>
<td>0.06</td>
<td>0.02</td>
<td>0.02</td>
<td>7.00</td>
</tr>
<tr>
<td>Feedback</td>
<td>TO</td>
<td>85.74</td>
<td>77.63</td>
<td>77.63</td>
<td>77.84</td>
<td>77.84</td>
<td>18.38</td>
<td>18.38</td>
<td>4.66</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>TO</td>
<td>TO</td>
<td>265.30</td>
<td>TO</td>
<td>TO</td>
<td>0.58</td>
<td>0.58</td>
<td>&gt;13.60</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>2.72</td>
<td>2.48</td>
<td>2.48</td>
<td>2.45</td>
<td>2.45</td>
<td>4.70</td>
<td>4.70</td>
<td></td>
</tr>
</tbody>
</table>
The two-phases approaches (e.g., bounded operation) can achieve a speedup of several orders of magnitude.
Conclusions

- RCS is a major bottleneck in HLS
  - Search Branch-and-bound approaches are promising for optimal resource-constrained scheduling

- Proposed an efficient two-phase B&B approach
  - Two-phase search space reduction
  - Partial-search heuristics
    - Bounded operation approach, non-chronological backtrack and search space speculation

- Successfully applied on various benchmarks with different resource constraints
  - Significant reduction in overall RCS efforts
Thank you !